

Parallelizing a State Exchange Strategy for Noncooperative Distributed NMPC[☆]

Jürgen Pannek

*Faculty of Aerospace Engineering
University of the Federal Armed Forces
85577 Munich, Germany*

Abstract

We consider a distributed non cooperative control setting in which systems are interconnected via state constraints. Each of these systems is governed by an agent which is responsible for exchanging information with its neighbours and computing a feedback law using a nonlinear model predictive controller to avoid violations of constraints. For this setting we present an algorithm which generates a parallelizable hierarchy among the systems. Moreover, we show both feasibility and stability of the closed loop using only abstract properties of this algorithm. To this end, we utilize a trajectory based stability result which we extend to the distributed setting.

Keywords: nonlinear model predictive control, stability, parallel algorithm

1. Introduction

Distributed control problems can arise either naturally, i.e. by a set of coupled systems which shall be controlled, see, e.g., Dold and Stursberg [4], or if a large problem is decomposed into smaller, again coupled problems, see Rawlings and Mayne [17, Chapter 10] or Scattolini [20] for an overview. In the latter case, the general idea is that smaller problems are solvable easier and faster which allows to even overcompensate the computational effort to coordinate these systems, cf. Richards and How [19, Section 7]. In either case, one distinguishes between cooperative control which features a centralized objective, and non cooperative control

[☆]This work was supported by the Leopoldina Fellowship Programme LPDS 2009-36.
Email address: juergen.pannek@gmail.com (Jürgen Pannek)

where the objectives of the systems are independent from each other. Using a centralized objective there are several possibilities to divide the optimization problem into subproblems. If suitable conditions hold then similar performance of the distributed control obtained from these subproblems and of the centralized control can be shown, see, e.g., Rawlings and Mayne [17, Chapter 10] or Giselsson and Rantzer [6].

Throughout this work we focus on the non cooperative control setting of systems driven by independent dynamics and control objectives, but coupled by constraints. For each system we impose an agent which exchanges state information with its neighbours and uses its local objective to compute a local control which satisfies the coupling constraints. For the computing task we focus on feedback design via nonlinear model predictive controller (NMPC) which minimizes the distance of the current state to the desired equilibrium over a finite time horizon. To show asymptotic stability of an NMPC closed loop, one often imposes additional stabilizing terminal constraints and costs, see, e.g., Keerthi and Gilbert [15] or Chen and Allgöwer [3] respectively. Since such terminal constraints may require long optimization horizons, we focus on the plain NMPC setting without those modifications. In the non distributed case, stability for such problems has been shown in Grüne et al. [11] whereas the distributed case is treated in Grüne and Worthmann [12] using the algorithm of Richards and How [18, 19].

Here, we first prove a stability idea outlined in Grüne and Worthmann [12] using the trajectory based setting of Grüne and Pannek [10, Chapter 7]. This proof allows us to reduce the horizon length in the distributed case while maintaining suboptimality estimates and stability like behavior of the closed loop. Secondly, since the computing time of the NMPC control law for each agent is not negligible, we present an algorithm which allows us to execute these computations in parallel using priority and deordering rules as well as a decision memory. From Rawlings and Mayne [17, Chapter 10] it is known that for the non cooperative control setting one can only expect to reach a Nash equilibrium. Although such a solution may be far from the optimal centralized solution, the closed loop solutions may still be stable and maintain the coupling constraints. For the proposed algorithm we present conditions under which feasibility of the closed loop is guaranteed and present necessary as well as sufficient conditions for asymptotic stability using only abstract properties of both the priority and the deordering rule. While here we focus on the plain NMPC case, we also outline how feasibility and stability results can be obtained using NMPC with terminal constraints or cost.

The paper is organized as follows: First, in Section 2 we formally define the problem under consideration for which we show different stability results for the

distributed case in Section 3. In the central Section 4, we present a covering algorithm which allows us to generate a hierarchy of agents and run the computations of each hierarchy level in parallel. Using this algorithm, we show necessary and sufficient conditions for feasibility and stability of the resulting closed loop and also how much parallelism can be achieved. Instead of a separated example section, we use an analytical example throughout the entire work to present the improvement of the stability result but also to illustrate the abstract functions used within the proposed algorithm in Section 4. Finally, we draw conclusions in Section 5 and present ideas for future research based on the presented work.

2. Setup and Preliminaries

Throughout this work we consider a set of nonlinear discrete time systems

$$x_p(n+1) = f_p(x_p(n), u_p(n)), \quad p \in \mathcal{P} := \{1, \dots, P\}, n \in \mathbb{N}_0 \quad (1)$$

with $x_p(n) \in X_p$ and $u_p(n) \in U_p$ and \mathbb{N}_0 denoting the set of natural numbers including zero. Here, X_p and U_p , $p \in \mathcal{P}$, are assumed to be arbitrary metric spaces denoting the state space and the set of admissible control values of the p -th system, respectively. The metrics to measure distances between two elements of X_p or of U_p are denoted by $d_{X_p} : X_p \times X_p \rightarrow \mathbb{R}_{\geq 0}$ and $d_{U_p} : U_p \times U_p \rightarrow \mathbb{R}_{\geq 0}$ where $\mathbb{R}_{\geq 0}$ denotes the positive reals including zero. In the following we denote the solution of a system p of (1) corresponding to the initial value $x_p(0) = x_p^0$ and the control sequence $u_p(k) \in U_p$, $k = 0, 1, 2, \dots$, by $x_p^u(k, x_p^0)$.

In order to define our goal we say that a continuous function $\alpha : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ is of class \mathcal{K}_∞ if it satisfies $\alpha(0) = 0$, is strictly increasing and unbounded. A continuous function $\gamma : \mathbb{R}_{\geq 0}^P \rightarrow \mathbb{R}_{\geq 0}$ is called a class \mathcal{K}_∞^P function if it satisfies $\gamma(0) = 0$, is strictly increasing in each component and is unbounded. A continuous function $\beta : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ is of class \mathcal{KL} if it is strictly decreasing in its second argument with $\lim_{t \rightarrow \infty} \beta(r, t) = 0$ for each $r > 0$ and satisfies $\beta(\cdot, t) \in \mathcal{K}_\infty$ for each $t \geq 0$. Moreover, $\mathcal{B}_r(x)$ denotes the open ball with center x and radius r and for arbitrary $x_1, x_2 \in X$ we denote the distance from x_1 to x_2 by $\|x_1\|_{x_2} = d_X(x_1, x_2)$.

For the set of systems (1) the overall system is given by

$$x(n+1) = f(x(n), u(n)), \quad n \in \mathbb{N}_0 \quad (2)$$

with state $x(n) = (x_1(n)^\top, \dots, x_P(n)^\top)^\top \in X = X_1 \times \dots \times X_P$ and control $u(n) = (u_1(n)^\top, \dots, u_P(n)^\top)^\top \in U = U_1 \times \dots \times U_P$. Now, our goal is to asymptotically stabilize system (2) at a desired equilibrium point $x^{\text{ref}} \in X$, i.e. to fulfill the following:

Definition 1. Let $x^{\text{ref}} \in X$ be an equilibrium for a system (2), i.e., there exists $u \in U$ such that $f(x^{\text{ref}}, u) = x^{\text{ref}}$. Then we say that x^{ref} is *locally asymptotically stable* if for a given control sequence $(u(n))_{n \in \mathbb{N}_0}$ there exist $r > 0$ and a function $\beta \in \mathcal{KL}$ such that the inequality

$$\|x^u(n, x^0)\|_{x^{\text{ref}}} \leq \beta(\|x^0\|_{x^{\text{ref}}}, n) \quad (3)$$

holds for all $x^0 \in \mathcal{B}_r(x^{\text{ref}})$ and all $n \in \mathbb{N}_0$.

Additionally, the solution $x^u(\cdot, x^0)$ shall satisfy state and control constraints. Throughout this work, we incorporate such constraints by considering suitable subsets of the overall state and control value space $\mathbb{X} \subset X$, $\mathbb{U} \subset U$ for system (2). As a result, systems (1) are coupled via the constraint sets \mathbb{X} and \mathbb{U} although the respective dynamics are decoupled. The following example illustrates this setting and will be used throughout this paper.

Example 2. Consider two cars attempting to cross a one lane bridge, i.e. one car has to wait, cf. Figure 1 for an illustration. Suppose the discrete time dynamics of the cars are given by

$$x_p(n+1) = x_p(n) + u_p(n)$$

with $u_p \in \mathbb{U}_p = \{-1, 0, 1\}^2$ and $x_p \in \mathbb{Z}^2$ for $p = 1, 2$. Since the cars shall not collide, we obtain the restriction

$$(x_{1,1}, x_{1,2})^\top \neq (x_{2,1}, x_{2,2})^\top. \quad (4)$$

The one lane bridge additionally imposes the constraints

$$x_{p,2} = 0 \text{ if } x_{p,1} = 0 \text{ for } p = 1, 2 \quad \text{and} \quad (5)$$

$$\begin{pmatrix} x_{1,1} + u_{1,1} \\ x_{1,2} + u_{1,2} \end{pmatrix} \neq \begin{pmatrix} x_{2,1} \\ x_{2,2} \end{pmatrix}, \quad \begin{pmatrix} x_{1,1} \\ x_{1,2} \end{pmatrix} \neq \begin{pmatrix} x_{2,1} + u_{2,1} \\ x_{2,2} + u_{2,2} \end{pmatrix} \quad (6)$$

which together form the set \mathbb{X} . Hence, the local set of admissible moves may depend on which car is allowed to drive first, cf. Figures 1a and 1b. Note that (5) is the only local constraint whereas — if they are considered — (4) induces an algebraic and (6) a neighbouring dynamic dependent coupling.

The purpose of this work is to show conditions under which stability of the overall systems can be guaranteed by using only local controllers. To this end, we

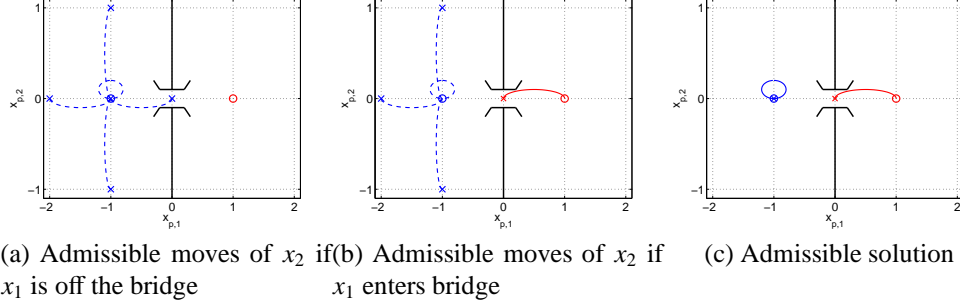


Figure 1: Admissible moves for initial conditions $x_1^0 = (1, 0)^\top$, $x_2^0 = (-1, 0)^\top$

impose an agent for each system $p \in \mathcal{P}$ to compute a suitable control sequence $u_p(\cdot) \in U_p^{\mathbb{N}_0} := \{u_p(k) \mid u_p(k) \in U_p \forall k \in \mathbb{N}_0\}$ and to exchange information with other agents in order to verify that the constraints $\mathbb{X} \subset X$, $\mathbb{U} \subset U$ are satisfied. Throughout this work each agent computes its control sequence via a nonlinear model predictive controller, a methodology which will be explained after Definition 6, below. In order to achieve asymptotic stability of the overall system (2) we develop a covering algorithm to coordinate all agents. The idea of this algorithm is the following: Since some subsystems impose constraints on each other, a priority rule is used to generate a hierarchy among the subsystems. As a result, agents which are on the same hierarchy level can compute their local optimal control in parallel while the hierarchy levels remain in serial. Additionally, a deordering rule is introduced to repeatedly verify if the hierarchy can be flattened, i.e. if more agents can work in parallel. For details on these rules we refer to Section 4.

Since we want to compute local controls u_p we must define the local constraints for each single system f_p , $p \in \mathcal{P}$. To this end, we “project” the constraint set \mathbb{X} to the state space of a subset of systems.

Definition 3. For an index set $\mathcal{I}_p = \{p_1, \dots, p_m\} \subset \mathcal{P}$ with $m \in \mathbb{N}$, $m \leq P$ and $p_i \neq p_j$ for all $i, j \in \{1, \dots, m\}$ the *set of partial states* is defined as $X_{\mathcal{I}_p} = X_{p_1} \times \dots \times X_{p_m}$ and we denote elements of $X_{\mathcal{I}_p}$ by $x_{\mathcal{I}_p} = (x_{p_1}, \dots, x_{p_m})$. Accordingly, the *partial state constraint set* is defined by

$$\mathbb{X}_{\mathcal{I}_p} := \{x_{\mathcal{I}_p} \in X_{\mathcal{I}_p} \mid \text{there is } \tilde{x} \in \mathbb{X} \text{ with } \tilde{x}_{p_i} = x_{p_i} \text{ for } i = 1, \dots, m\}.$$

In case of Example 2, Definition 3 basically means that only those constraints induced by the neighbours contained in \mathcal{I}_p have to be considered, i.e. if agent

$p = 1$ ignores agent $p = 2$, then only constraint (5) has to be fulfilled. In the general case, this task can be accomplished by using information on neighbouring systems which are available to an agent and allow to generate a hierarchy among the agents. Here, we assume that this information can be exchanged repeatedly in between two time instants n and $n + 1$.

Throughout this work we consider changing network topologies, i.e. the sets of neighbours at time instants n and $n + 1$ may differ, see Figure 2 for an illustration.

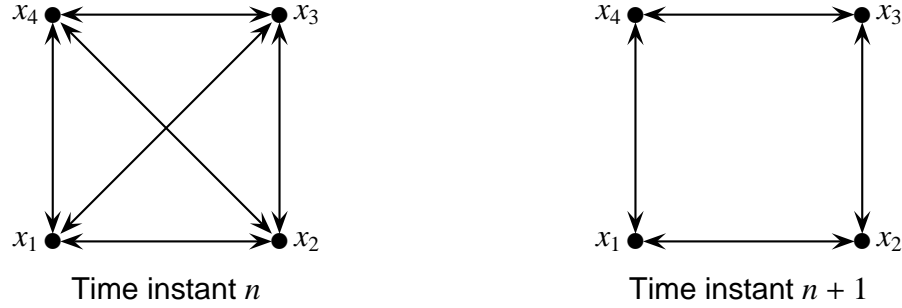


Figure 2: Possible sequence of communication graphs at time instants n and $n + 1$

Additionally, we allow the case that even if neighbouring information of a system $q \in \mathcal{P} \setminus \{p\}$ is known to an agent $p \in \mathcal{P}$, agent p ignores that information if his hierarchy level is higher than the level of the neighbour. Consequently, the dependency graph which results from this hierarchy may differ from the communication graph as illustrated in Figure 3.



Figure 3: Possible difference of communication and dependency graph

Moreover, as sent information may be delayed or even lost, we want to allow for considering old information on neighbours and variable lengths of this information. Note that the latter may also allow agents to skip recomputations of controls. Introducing N_q as the length and n_q as the time instant at which neighbour

q has computed the state sequence $(x_q^{n_q}(0), \dots, x_q^{n_q}(N_q))$, we define the exchanged neighbouring information as follows:

Definition 4. Suppose that at time instant $n \in \mathbb{N}_0$ agent p knows the state sequences $x_q^{n_q}(\cdot) = (x_q^{n_q}(0), \dots, x_q^{n_q}(N_q))$, $N_q \in \mathbb{N}_0$, computed at time instant $n_q \leq n$ for a given *neighbouring index set* $\mathcal{I}_p(n)$, that is $q \in \mathcal{I}_p(n)$ with $p \notin \mathcal{I}_p(n)$. We define the *neighbouring information* as

$$I_p(n) = \{(q, n_q, N_q, x_q^{n_q}(\cdot)) \mid q \in \mathcal{I}_p(n)\}$$

being an element of the set $\mathbb{I}_p = 2^{\mathcal{Q}}$ with $\mathcal{Q} = (\mathcal{P} \setminus \{p\}) \times \mathbb{N}_0 \times \mathbb{N}_0 \times X^{\mathbb{N}}$.

Knowing the states of neighbouring systems for a certain time period, we can define the index set used within the “projection” of the constraint set \mathbb{X} .

Definition 5. For a given time instant $n \in \mathbb{N}_0$ and an agent $p \in \mathcal{P}$ with neighbouring information $I_p(n)$, we call the set of systems $q \in \mathcal{I}_p(n) \setminus \{p\}$ which are imposing constraints on system p at time instant $n + k \in \mathbb{N}_0$, $k \geq 0$ *neighbouring prediction index set*. This set is given by

$$\mathcal{I}_p(n, k) = \{q \in \mathcal{I}_p(n) \setminus \{p\} \mid n + k \leq n_q + N_q\}.$$

Similar to possible moves of the cars in Example 2, we can use the partial state constraint set connected to neighbouring information available to an agent and define the set of admissible controls from which the control sequence $u_p(\cdot)$ can be chosen, cf. Figures 1a, 1b.

Definition 6. Given a time instant $n \in \mathbb{N}_0$ and an agent $p \in \mathcal{P}$ with initial value x_p^0 and neighbouring information $I_p(n)$, we define the *set of admissible control sequences* for system p at time instant n as

$$\begin{aligned} \mathbb{U}_p^{\text{ad}}(n, x_p^0, I_p(n)) = \{u_p(\cdot) \in U_p^{\mathbb{N}_0} \mid \text{for all } k = 0, 1, \dots \text{ we have } u_p(k) \in \mathbb{U}_p \text{ and} \\ (x_p^u(k, x_p^0), (x_q^{n_q}(k + n - n_q))_{I_p(n, k)}) \in \mathbb{X}_{\{p\} \cup \mathcal{I}_p(n, k)}\}. \end{aligned}$$

Using an NMPC algorithm is one possibility to compute a control from the set of admissible controls. In particular, the method tries to approximate a control sequence such that the functional

$$J_p^\infty(x_p^0, u_p) = \sum_{k=0}^{\infty} \ell_p(x_p^u(k, x_p^0), u_p(k)) \quad (7)$$

is minimized over all admissible control sequences, that is sequences $u_p(\cdot)$ with $u_p(k) = u_p^*(0)$ for all $k \in \mathbb{N}_0$ with $u_p^* \in \mathbb{U}_p^{\text{ad}}(k, x_p^u(k, x_p^0), I_p(k))$. Here, the function ℓ_p is a stage cost function penalizing both the distance of the state to the desired equilibrium and the used control. A popular choice for this function is $\ell_p(x_p, u_p) = \|x_p\|_{x_p^{\text{ref}}} + \lambda \|u_p\|_{u_p^{\text{ref}}}$ with weighting parameter $\lambda > 0$.

Computing a control minimizing (7) is, in general, computational intractable. To circumvent this issue the NMPC algorithm uses the truncated cost functional

$$J_p^{N_p}(x_p^0, u_p) = \sum_{k=0}^{N_p-1} \ell_p(x_p^u(k, x_p^0), u_p(k)) \quad (8)$$

with finite horizon of length N_p and initial value x_p^0 . Hence, a finite minimizing control sequence $u_p^* \in \mathbb{U}_p^{N_p, \text{ad}}(n, x_p^0, I_p(n))$ is computed with

$$\mathbb{U}_p^{N_p, \text{ad}}(n, x_p^0, I_p(n)) = \{u_p(\cdot) \in U_p^{N_p} \mid \text{for all } k = 0, \dots, N_p \text{ we have } u_p(k) \in \mathbb{U}_p \text{ and } (x_p^u(k, x_p^0), (x_q^{n_q}(k + n - n_q))_{I_p(n, k)}) \in \mathbb{X}_{\{p\} \cup I_p(n, k)}\}.$$

In the following we assume that a minimizing control sequence exists and denote the corresponding optimal value function by

$$V_p^{N_p}(x_p(n), I_p(n)) = \min_{u_p \in \mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), I_p(n))} J_p^{N_p}(x_p(n), u_p)$$

where the minimizing control sequence is given by

$$u_p^* = \underset{u_p \in \mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), I_p(n))}{\operatorname{argmin}} J_p^{N_p}(x_p(n), u_p).$$

Here, the argmin operator is used in the following sense: given a map $a : U \rightarrow \mathbb{R}$, a nonempty subset $\widetilde{U} \subseteq U$ and a value $u^* \in \widetilde{U}$ we write $u^* = \operatorname{argmin}_{u \in \widetilde{U}} a(u)$ if and only if $a(u^*) = \min_{u \in \widetilde{U}} a(u)$ holds. Note that we do not require uniqueness of the minimizer u^* . In case of uniqueness the argmin operator can be understood as an assignment, otherwise it is just a convenient way of writing “ u^* minimizes $a(u)$ ”.

Having obtained a minimizing sequence $u_p^*(\cdot)$, only the first element $u_p^*(0)$ of the control sequence is implemented. Then the entire problem is shifted forward in time by one time instant and both a new initial value and neighbouring information need to be obtained. Applying this method iteratively results in a feedback law which assigns the first element of the minimizing control sequence $u_p^*(\cdot)$ to the

current state of the p -th system $x_p(n)$ and the neighbouring information $I_p(n)$ of the corresponding agent, i.e. a map

$$\mu_p^{N_p} : (x_p(n), I_p(n)) \mapsto u_p^*(0). \quad (9)$$

Accordingly, the closed loop solution of the p -th system is given by

$$x_p(n+1) = f(x_p(n), \mu_p^{N_p}(x_p(n), I_p(n))) \quad \text{with} \quad x_p(0) = x_p^0. \quad (10)$$

Using this setting, we first show conditions which guarantee asymptotic stability of the closed loop for local controllers.

3. Stability

While commonly endpoint constraints or a Lyapunov function type endpoint weight are used to ensure stability of the closed loop, see, e.g., the articles of Keerthi and Gilbert [15], Chen and Allgöwer [3], Jadbabaie and Hauser [14] and Graichen and Kugi [7], we consider the plain NMPC version without these modifications. In order to guarantee stability in this case, we use the “relaxed” version of the dynamic programming principle, cf. Lincoln and Rantzer [16]. In particular, one can show asymptotic stability of (2) in a trajectory based setting using a relaxed Lyapunov condition, see Grüne and Pannek [10, Proposition 7.6]. Note that this stability result requires a centralized setting and the horizons to satisfy $N_p = N$ for all $p \in \mathcal{P}$. Hence, for the overall system (2) we denote the combined stage costs by $\ell(x(n), u(n))$, the finite and infinite cost functional by $J^N(x^0, u)$, $J^\infty(x^0, u)$ and the corresponding combined value functions by $V^N(x(n))$, $V^\infty(x(n))$ which allows us to apply the stability result of Grüne and Pannek [10, Proposition 7.6]:

Proposition 7. *Consider a feedback law $\mu^N : \mathbb{X} \rightarrow \mathbb{U}$ and the closed loop trajectory $x(\cdot)$ of (2) with control $u = \mu^N$ and initial values $x(0) \in \mathbb{X}$ to be given. If the optimal value function $V^N : \mathbb{X} \rightarrow \mathbb{R}_{\geq 0}$ satisfies*

$$V^N(x(n)) \geq V^N(f(x(n), \mu^N(x(n))) + \alpha \ell(x(n), \mu^N(x(n))) \quad (11)$$

for some $\alpha \in (0, 1]$ and all $n \in \mathbb{N}_0$, then

$$\alpha V^\infty(x(n)) \leq \alpha J^\infty(x(n), \mu^N) \leq V^N(x(n)) \leq V^\infty(x(n)) \quad (12)$$

holds for all $n \in \mathbb{N}_0$.

If, in addition, there exist $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{K}_\infty$ such that

$$\alpha_1(\|x\|_{x^{\text{ref}}}) \leq V^N(x) \leq \alpha_2(\|x\|_{x^{\text{ref}}}) \quad \text{and} \quad \ell(x, u) \geq \alpha_3(\|x\|_{x^{\text{ref}}}) \quad (13)$$

holds for all $x(n) \in \mathbb{X}$ with $n \in \mathbb{N}_0$, then there exists a function $\beta \in \mathcal{KL}$ which only depends on $\alpha_1, \alpha_2, \alpha_3$ and α such that the inequality

$$\|x(n)\|_{x^{\text{ref}}} \leq \beta(\|x(0)\|_{x^{\text{ref}}}, n) \quad (14)$$

holds for all $n \in \mathbb{N}_0$, i.e., x behaves like a trajectory of an asymptotically stable system.

The key assumption in Proposition 7 is the relaxed Lyapunov–inequality (11) in which α can be interpreted as a lower bound for the rate of convergence. From the literature, it is well-known that this condition is satisfied for sufficiently long horizons N , cf. Jadbabaie and Hauser [14], Grimm et al. [8] or Alamir and Bornard [1], and that a suitable N may be computed via methods described in Grüne and Pannek [10, Chapter 7] or Giselsson [5].

Now we consider a distributed setting of Proposition 7 using compositions to combine of the set of systems (1). The idea of such compositions is to introduce a weighting among the subsystems which in our further analysis will allow for increases of costs along the closed loop for some subsystems.

Proposition 8. Consider feedback laws $\mu_p^N : \mathbb{X}_p \times \mathbb{I}_p \rightarrow \mathbb{U}_p$ and closed loop trajectories $x_p(\cdot)$ of (10) with initial values $x_p(0) \in \mathbb{X}_p$ to be given. If the optimal value functions $V_p^N : \mathbb{X}_p \rightarrow \mathbb{R}_{\geq 0}$ satisfy

$$V_p^N(x_p(n)) \geq V_p^N(f_p(x_p(n), \mu_p^N(x_p(n), I_p(n)))) + \alpha \ell_p(x_p(n), \mu_p^N(x_p(n), I_p(n))) \quad (15)$$

for some $\alpha \in (0, 1]$ and all $n \in \mathbb{N}_0$, then for any weighting function $\gamma : \mathbb{R}^P \rightarrow \mathbb{R}_{\geq 0}$, $\gamma(x) = \gamma^\top x$, $\gamma_i \in \mathbb{R}_{>0}$ we have that (12) holds for all $n \in \mathbb{N}_0$ with

$$V^N(x) := \gamma((V_1^N(x_1), \dots, V_P^N(x_P))^\top) \quad \text{and} \quad \ell(x, u) := \gamma((\ell_1(x_1, u_1), \dots, \ell_P(x_P, u_P))^\top).$$

If, in addition, for every $p \in \mathcal{P}$ there exist $\alpha_1^p, \alpha_2^p, \alpha_3^p \in \mathcal{K}_\infty$ such that

$$\alpha_1^p(\|x_p\|_{x_p^{\text{ref}}}) \leq V_p^N(x) \leq \alpha_2^p(\|x_p\|_{x_p^{\text{ref}}}) \quad \text{and} \quad \ell_p(x_p, u_p) \geq \alpha_3^p(\|x_p\|_{x_p^{\text{ref}}}) \quad (16)$$

holds for all $x_p(n) \in \mathbb{X}$ with $n \in \mathbb{N}_0$, then there exists a function $\beta \in \mathcal{KL}$ which only depends on γ , α and all $\alpha_1^p, \alpha_2^p, \alpha_3^p$, $p \in \mathcal{P}$, such that (14) holds for all $n \in \mathbb{N}_0$.

Proof. Defining the abbreviations $V_{\mathcal{P}}^N(x(n)) := (V_1^N(x_1(n)), \dots, V_P^N(x_P(n)))^\top$ and $\ell_{\mathcal{P}}(x(n), \mu^N(x(n), I_{\mathcal{P}}(n))) := (\ell_1(x_1(n), \mu_1^N(x_1(n), I_1(n))), \dots, \ell_P(x_P(n), \mu_P^N(x_P(n), I_P(n))))^\top$ we combine all inequalities (15) for $p \in \mathcal{P}$ and obtain

$$\gamma(V_{\mathcal{P}}^N(x(n))) \geq \gamma(V_{\mathcal{P}}^N(x(n+1))) + \alpha \gamma(\ell_{\mathcal{P}}(x(n), \mu^N(x(n), I_{\mathcal{P}}(n)))). \quad (17)$$

using linearity of γ . Now we use the definition of V^N and ℓ which gives us (11). Hence, (12) follows directly from Proposition 7. Similarly, (14) follows by definition of V^N and ℓ which together with $\alpha_i(r) := \gamma((\alpha_i^1(r), \dots, \alpha_i^P(r))^\top)$, $i = 1, 2, 3$, and again Proposition 7 shows the assertion. \square

Certainly, condition (15) would be desirable since it guarantees a decrease in V_p^N for each $p \in \mathcal{P}$. In practice, however, one would usually expect V_p^N to decrease for some $p \in \mathcal{P}$ while it increases for others as shown in the following example:

Example 9. Consider the setting of Example 2 where we suppose that each agent $p = 1, 2$ is optimizing using its running costs $\ell_p(x_p, u_p) = \|x_p - x_p^{\text{ref}}\|_2^2$ with $x_1^{\text{ref}} = (-2, 0)^\top$ and $x_2^{\text{ref}} = (2, 0)^\top$ and complete neighbouring information for initial conditions $x_1^0 = (1, 0)^\top$, $x_2^0 = (-1, 0)^\top$.

Due to the constraints (4)–(6) one car has to wait before entering the bridge, cf. Figure 1c, and even has to move aside as shown in Figure 4a. Without loss of generality we assume that system $p = 2$ moves aside. Although the optimal con-

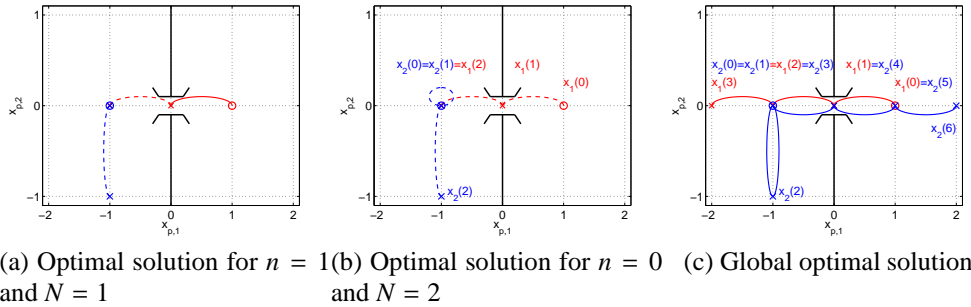


Figure 4: Optimal open and closed loop trajectories

trol for agent $p = 2$ for any $N \geq 2$ is identical to the global optimal control, cf. Figures 4b and 4c, we obtain $V_2^N(x_2(0), I_2(0)) \leq V_2^N(x_2(1), I_2(1))$ for $N = 2$ and

$N = 3$. Hence, we cannot guarantee (15) to hold for these N although the closed loop is stable. For larger values of N , however, we obtain

$$\begin{aligned} V_2^N(x_2(0), I_2(0)) &= \begin{cases} 37 & \text{if } N = 4 \\ 41 & \text{if } N = 5 \\ 42 & \text{if } N \geq 6 \end{cases} \quad \text{and} \quad \ell_2(x_2(0), \mu_2^N(x_2(0), I_2(0))) = 9, \\ V_2^N(x_2(1), I_2(1)) &= \begin{cases} 32 & \text{if } N = 4 \\ 33 & \text{if } N \geq 5 \end{cases} \quad \text{and} \quad \ell_2(x_2(1), \mu_2^N(x_2(1), I_2(1))) = 9, \end{aligned}$$

$$\begin{aligned} (V_2^N(x_2(n), I_2(n)))_{n=2, \dots, 5, N \geq 4} &= (24, 10, 5, 1) \quad \text{and} \\ (\ell_2(x_2(n), \mu_2^N(x_2(n), I_2(n))))_{n=2, \dots, 5, N \geq 4} &= (10, 9, 4, 1) \end{aligned}$$

and $V_2^N(x_2(n), I_2(n)) = \ell_2(x_2(n), \mu_2^N(x_2(n), I_2(n))) = 0$ if $N \geq 4$ and $n \geq 6$. Accordingly, the largest values α such that (15) holds are $\alpha = 5/9$ if $N = 4$, $\alpha = 8/9$ if $N = 5$ and $\alpha = 1$ if $N \geq 6$. Hence, we can use Proposition 8 to conclude asymptotic stability if $N \geq 4$.

In an indeep analysis, Grüne and Worthmann [12, Theorem 5.3] have shown conditions such that for the serial case using the algorithm of Richards and How [18, 19] inequalities (15) and (16) hold. Note that although the setting within the articles of Richards and How [18, 19] is for one based on linear dynamics and secondly explicitly includes perturbations in the models, this algorithm can also be used in a nonlinear setting, cf. Grüne and Worthmann [12, Proposition 3.2].

Taking a closer look at the proof of Proposition 8 we see that in fact conditions (15) are only required to guarantee (17) to hold. Note that while condition (15) requires a decrease in V_p^N for each $p \in \mathcal{P}$, in (17) it suffices that V_p^N , $p \in \mathcal{P}$ is decreasing under a map γ . Moreover, we only require $\gamma \in \mathcal{K}_\infty^P$ in the remainder of the proof of Proposition 8 to guarantee $\alpha_i \in \mathcal{K}_\infty$, $i = 1, 2, 3$. Accordingly, we obtain the following more general result:

Proposition 10. *Consider feedback laws $\mu_p^N : \mathbb{X}_p \times \mathbb{I}_p \rightarrow \mathbb{U}_p$ and closed loop trajectories $x_p(\cdot)$ of (10) with initial values $x_p(0) \in \mathbb{X}_p$ to be given. If the optimal value functions $V_p^N : \mathbb{X}_p \rightarrow \mathbb{R}_{\geq 0}$ satisfy (17) for some $\alpha \in (0, 1]$, $\gamma \in \mathcal{K}_\infty^P$ and all $n \in \mathbb{N}_0$, then (12) holds for all $n \in \mathbb{N}_0$ with V^N and ℓ defined as in Proposition 8.*

If, in addition, for every $p \in \mathcal{P}$ there exist $\alpha_1^p, \alpha_2^p, \alpha_3^p \in \mathcal{K}_\infty$ such that (16) holds for all $x_p(n) \in \mathbb{X}$ with $n \in \mathbb{N}_0$, then there exists a function $\beta \in \mathcal{KL}$ which only depends on γ , α and all $\alpha_1^p, \alpha_2^p, \alpha_3^p$, $p \in \mathcal{P}$, such that (14) holds for all $n \in \mathbb{N}_0$.

Proof. Follows directly from the proof of Proposition 8. \square

The conclusion that can be drawn from Proposition 10 is that the weighting function γ may allow us to partially violate condition (15). Since (15) is typically fulfilled if the horizon N_p is large enough, a good choice of γ may reduce the horizon length N_p as we will see in the following example:

Example 11. Consider Example 9 and suppose γ to be the 1–norm, then we obtain for $N = 2$

$$\begin{aligned} \left(\sum_{p=1}^2 V_p^N(x_p(n), I_p(n)) \right)_{n=0, \dots, 5} &= (31, 24, 20, 13, 5, 1) \\ \left(\sum_{p=1}^2 \ell_p(x_p(n), \mu_p^N(x_p(n), I_p(n))) \right)_{n=0, \dots, 5} &= (18, 13, 11, 9, 4, 1) \end{aligned}$$

and $\sum_{p=1}^2 V_p^N(x_p(n), I_p(n)) = \sum_{p=1}^2 \ell_p(x_p(n), \mu_p^N(x_p(n), I_p(n))) = 0$ for $n \geq 6$. Hence, (17) holds with $\alpha = 4/13$ and we obtain asymptotic stability of the closed loop by Proposition 10. Since $\alpha > 0$ holds for all $N \geq 2$ this example illustrates the advantage of considering condition (17) instead of (15).

As outlined before Proposition 10, under certain conditions the algorithm of Richards and How [18, 19] can be applied to generate solutions such that (15) and (16) hold. However, the nature of this algorithm is serial, that is while one agent $p \in \mathcal{P}$ is computing its control, all other agents $q \in \mathcal{P} \setminus \{p\}$ have to wait until agent p finished computing. Hence, if the number of systems P is large, such an algorithm may cause rather long waiting times, a feature which may be unwanted if fast sampling is used. Still, as noted in Richards and How [19, Section 7], due to its decentralized nature the dimension of each problem is significantly smaller and hence the algorithm reduces the numerical effort compared to a centralized solution considerably.

Apart from the serial nature, the algorithm of Richards and How requires accessibility to the full neighbouring information, i.e. a full communication graph. Additionally, an agent $p \in \mathcal{P}$ always uses the latest available neighbouring information to compute a minimizing control u_p^* which results in a full dependency graph. While the latter condition on the dependency graph may be relaxed easily, it is a complex task to obtain a parallel algorithm and to relax the requirement of a full communication graph.

4. The covering algorithm

In this section we provide a covering algorithm which is a modification of the Algorithm of Richards and How [18, 19] and allows us to run the agents $p \in \mathcal{P}$ in parallel if they are independent from one another. Unfortunately, working in a parallel distributed setting omits the use of standard techniques from optimization such as first and second order information of the cost functional and the constraints for the interlink between systems to search for optimal controls.

To circumvent this deficiency we introduce abstract maps $\Pi, \Theta : 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$ which denote priority and deordering rules, see Haupt [13] and Bäckström [2]. The aim of this section is to show how much parallelism can be expected using the algorithm we propose next and the basic properties of Π, Θ being a permutation and a self concatenation mapping respectively.

The structural layout of the algorithm we present now is closely related to the NMPC algorithm outlined in Section 2:

Algorithm 12. Set lists $\mathcal{P}_1 := (1, \dots, P)$ and $\mathcal{P}_p := \emptyset$ for $p = 2, \dots, P$, $n := 0$ and $I_p(n) := \emptyset$ for $p = 1, \dots, P$.

1. Obtain new measurements $x_p(n)$ for $p \in \mathcal{P}$.
- 2a. (*Decision memory and deordering rule*) For i from 2 to P do
 - For j from 1 to $\#\mathcal{P}_i$ do
 - (i) Set $\mathcal{I}_{p_j}(n) := \Theta(\mathcal{I}_{p_j}(n)) \subsetneq \mathcal{I}_{p_j}(n)$
 - (ii) If $\mathcal{I}_{p_j}(n) = \emptyset$, then remove p_j from \mathcal{P}_i and set $\mathcal{P}_1 := (\mathcal{P}_1, p_j)$
 Else: If $\tilde{m} = \min_{k \in \mathcal{P}_m, p_k \in \mathcal{I}_{p_j}(n)} m < i$ holds, then remove p_j from \mathcal{P}_i and set $\mathcal{P}_{\tilde{m}} := (\mathcal{P}_{\tilde{m}}, p_j)$
- 2b. Compute a control $u_p^*(\cdot)$ minimizing (7) or (8) with $x_p^0 = x_p(n)$ and send information to all agents $q \in \{q \in \mathcal{P} \mid q \in \mathcal{P}_j, p \in \mathcal{P}_i \text{ and } j \geq i\}$ for $p \in \mathcal{P}$ in parallel
- 2c. (*Priority rule*) For i from 1 to P do
 - (i) If $\#\mathcal{P}_i \in \{0, 1\}$, goto Step 3.
 Else: Sort index list by setting $\mathcal{P}_i := \Pi(\mathcal{P}_i)$
 - (ii) For j from 2 to $\#\mathcal{P}_i$ do

If system p_j violates constraints imposed by systems $p_k, k < j$, then set $\mathcal{P}_{i+1} := (\mathcal{P}_{i+1}, j)$ and $\mathcal{I}_j(n) := \mathcal{I}_j(n) \cup \{p_k \in \mathcal{P}_i \setminus \mathcal{P}_{i+1} \mid p_k, k < j, \text{ induces constraints violated by system } p_j\}$ and set $\mathcal{P}_i := \mathcal{P}_i \setminus \mathcal{P}_{i+1}$
 - (iii) Compute a control $u_p^*(\cdot, \mathcal{I}_p(n))$ minimizing (7) or (8) for all $p \in \mathcal{P}_{i+1}$ in parallel and send information to all agents $q \in \{q \in \mathcal{P} \mid q \in \mathcal{P}_j, j \geq i\}$

3. Implement $\mu_p^{N_p}(x_p(n), I_p(n)) := u_p^*(0)$, set $n := n + 1$ and goto Step 1.

The general idea of the algorithm is to first generate priority lists \mathcal{P}_i of the systems according to the rule Π and according to their interconnection with other systems, cf. Step 2c, – just as the right-before-left rule in street traffic or the search direction in optimization methods. Secondly, these lists are used to remember earlier decisions which avoids generating periodic behaviour. This part of the algorithm, contained in Step 2a, is inspired by Bland’s rule and the lexicographic ordering method used in the simplex algorithm to cope with degeneracy. Last, the deordering rule Θ which is used together with the memory in Step 2a offers a possibility to break up earlier decisions. Proceeding this way avoids blockages and reduces both the number of priority lists and thereby the numerical effort to compute the control sequences.

We like to mention that Algorithm 12 can be extended to an iterative computation of the controls u_p^* , $p \in \mathcal{P}$. To this end only a few steps within the optimization method used to solve the problems of Steps 2b and 2c(iii) are performed. Additionally a second loop containing Steps 2b and 2c is introduced which is terminated if some stopping criterion like the suboptimality based criterion given in Grüne and Pannek [9] is satisfied. Note that the algorithm also allows us to stop agents during such an iterative computation, i.e. if (15) is satisfied for some $\alpha \geq \bar{\alpha} \in (0, 1)$. Since we allowed for using old and even outdated information in Definition 4, the algorithm even allows to block any computations of some agents for a certain period depending on the length of an agents prediction without compromising feasibility.

Given Algorithm 12, we first consider the question whether a feasible feedback $\mu_p^{N_p}$ can be computed via Algorithm 12:

Theorem 13. *Assume a feasible initial value $x_0 \in \mathbb{X}$ for system (2) to be given. Suppose that for all $p \in \mathcal{P}$ and all $n \in \mathbb{N}_0$ we have that the sets of admissible controls $\mathbb{U}_p^{ad}(n, x_p(n), I_p(n))$ in case of cost functional (7) or $\mathbb{U}_p^{N_p, ad}(n, x_p(n), I_p(n))$ in case of cost functional (8) in Steps 2b and 2c(iii) are not empty, then the closed loop solutions (10) satisfy $x(n) = (x_1(n)^\top, \dots, x_P(n)^\top)^\top \in \mathbb{X}$.*

Proof. Using $x_0 \in \mathbb{X}$ and $\mathbb{U}_p^{ad}(0, x_p(0), I_p(0)) \neq \emptyset$ for all $p \in \mathcal{P}$ in case of cost functional (7) or $\mathbb{U}_p^{N_p, ad}(0, x_p(0), I_p(0)) \neq \emptyset$ for all $p \in \mathcal{P}$ in case of cost functional (8), we obtain from Steps 2b and 2c(iii) that optimal controls $u_p^*(\cdot, I_p(0))$ exist for all $p \in \mathcal{P}$. Hence, by definition of the closed loop in (10) and Step 3 we obtain that $x(1) = (x_1(1)^\top, \dots, x_P(1)^\top)^\top \in \mathbb{X}$ holds. Applying the same argumentation inductively for all $n \in \mathbb{N}_0$ the assertion follows. \square

Before showing results for Algorithm 12 together with general priority and deordering rules Π , Θ , we like to illustrate both rules using the example outlined in Section 3. The idea of the priority rule is straight forward. In fact, we have already used it in Example 9 to solve the blockage in the very first step:

Example 14. Again consider Example 9. Due to the constraint sets \mathbb{X} , \mathbb{U} and the dynamics of the systems f_p , one of the agents p has to move aside first to let the system of the other agent pass by before it can proceed towards its desired equilibrium. Putting priority of agent $p = 1$ into a mathematical form, we see that Π can be implemented as a lexicographic ordering, that is a list \mathcal{L} is mapped to its minimal permutation with respect to the dictionary ordering $<^d$ induced by the total orderings $\{<_1, \dots, <_m\}$ where m is the length of the list \mathcal{L} and $<_i, i = 1, \dots, m$ is the usual ordering $<$ of the natural numbers \mathbb{N} .

Apart from the lexicographic ordering, also other heuristics like the greedy heuristic might be used. It is not clear how the priority rule should be chosen in a nonlinear setting, and throughout this work we will not focus on this question but instead concentrate on general properties of Algorithm 12.

The idea of the deordering rule Θ is more involved as it may interfere with the idea of keeping track of earlier decisions. The purpose of this rule is to reduce the number of the priority lists since Step 2c of Algorithm 12 is a serial call for all lists \mathcal{P}_i . Accordingly, agents $p \in \mathcal{P}_{i+1}$ always have to wait until all agents $p \in \mathcal{P}_i$ have finished computing, a fact we wish to avoid. Note that this serial nature is independent from the parallel computation of control sequences $u_p^*, p \in \mathcal{P}_i$. Using the deordering rule Θ allows us to “test” whether a system $p \in \mathcal{P}_i$ still interferes with all systems $p \in \mathcal{P}_k, k < i$, or if it can be inserted into a different priority list $\mathcal{P}_k, k < i$, causing the number of lists and hence the number of non parallel steps to shrink. Yet even if system p cannot be inserted in a different priority list, applying the deordering rule might still result in reducing the size of the neighbouring index set $\mathcal{I}_p(n)$. If this is the case, then the number of constraints of system p is reduced which in turn reduces the numerical effort to compute the control sequence u_p^* .

Example 15. Consider once more Example 9 with $\Theta(\mathcal{P}) = \emptyset$. Applying Algorithm 12 we obtain that u_2^* depends on the solution of system $p = 1$ for $n \in \{0, 1\}$ only whereas for all $n \geq 2$ both problems can be solved in parallel.

Turning towards the central point of this section, we now analyze how much parallelism is possible even if we do not know the exact sorting and testing operators Π, Θ . Based on conditions on the priority lists \mathcal{P}_i our first result shows in which case all agents can compute their controls independently from each other:

Lemma 16. Suppose that for given systems (1), maps $\Pi, \Theta : 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$ and $n \in \mathbb{N}_0$ we have that $\mathcal{P}_2 = \emptyset$ holds in Step 2c(i) of Algorithm 12. Then every agent $p \in \mathcal{P}$ can compute its control sequence independently of all other agents $q \in \mathcal{P} \setminus \{p\}$.

Proof. Since $\mathcal{P}_2 = \emptyset$ Step 2c(ii) of Algorithm 12 guarantees that there are no systems $p_1, p_2 \in \mathcal{P}_1$, $p_1 \neq p_2$, such that p_1 induces a constraint on p_2 which is violated by p_2 , i.e. $\mathcal{I}_p(n) = \emptyset$ for all $p \in \mathcal{P}$. Hence, for each agent $p \in \mathcal{P}$ the set of admissible controls simplifies to

$$\mathbb{U}_p^{\text{ad}}(n, x_p^0, I_p(n)) = \{u_p(\cdot) \in U_p^{\mathbb{N}_0} \mid u_p(k) \in \mathbb{U}_p \text{ and } x_p^u(k, x_p^0) \in \mathbb{X}_p \text{ for all } k \in \mathbb{N}_0\}$$

if cost functional (7) or

$$\begin{aligned} \mathbb{U}_p^{N_p, \text{ad}}(n, x_p^0, I_p(n)) &= \{u_p(\cdot) \in U_p^{N_p} \mid u_p(k) \in \mathbb{U}_p \text{ and } x_p^u(k, x_p^0) \in \mathbb{X}_p \\ &\quad \text{for all } k \in \{0, \dots, N_p\}\} \end{aligned}$$

if cost functional (8) is considered with $x_p^0 = x_p(n)$ showing the assertion. \square

Using the self-concatenation property of the map Θ , we can also show that under certain conditions the priority lists show dependency of agents:

Lemma 17. Consider systems (1), $P \geq 2$ to be given. Suppose that applying Algorithm 12 for given maps $\Pi, \Theta : 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$ we have that $\mathcal{P}_i \neq \emptyset$ with $i \geq 2$ holds for some $n \geq \bar{n}$ and all $\bar{n} \in \mathbb{N}_0$. Then for each system $p \in \mathcal{P}_i$ there exists at least one system $q \in \mathcal{P}_j$, $j < i$ such that $q \in \mathcal{I}_p(n)$. Moreover, in case cost functional (7) is used, we have

$$u_p^* = \underset{u_p \in \mathbb{U}_p^{\text{ad}}(n, x_p(n), \emptyset)}{\operatorname{argmin}} J_p^\infty(x_p(n), u_p) \notin \mathbb{U}_p^{\text{ad}}(n, x_p(n), I_p(n)) \subsetneq \mathbb{U}_p^{\text{ad}}(n, x_p(n), \emptyset)$$

and in case of cost functional (8) we have

$$u_p^* = \underset{u_p \in \mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), \emptyset)}{\operatorname{argmin}} J_p^{N_p}(x_p(n), u_p) \notin \mathbb{U}_p^{\text{ad}}(n, x_p(n), I_p(n)) \subsetneq \mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), \emptyset).$$

Proof. Suppose that $\mathcal{P}_i \neq \emptyset$ with $i \geq 2$ holds for some $n \geq \bar{n}$ and all $\bar{n} \in \mathbb{N}_0$ and fix $p \in \mathcal{P}_i$ arbitrarily. Suppose furthermore that there exists no $q \in \mathcal{P}_j$, $j < i$ such that $q \in \mathcal{I}_p(n)$ holds. Then, by the deordering rule Θ and Step 2a(i) we obtain that there exists $\bar{n} \in \mathbb{N}_0$ such that $\mathcal{P}_i = \emptyset$ for all $n \geq \bar{n}$ contradicting our assumption. Hence, since $p \in \mathcal{P}_i$ was chosen arbitrarily, we obtain that for each $p \in \mathcal{P}_i$ there exists a system $q \in \mathcal{P}_j$, $j < i$ such that $q \in \mathcal{I}_p(n)$ holds.

Now, due to Step 2c(ii) and the fact that there exists a system $q \in \mathcal{I}_p(n)$ imposing constraints on system p which are violated if $q \notin \mathcal{I}_p(n)$ the assertion for both cost functionals (7) and (8) follows. \square

Now we can use Lemma 17 to answer the question under which conditions asymptotic stability can be shown. In particular, we first prove a necessary condition for asymptotic stability of (2).

Theorem 18. *Consider systems (1), $P \geq 2$ to be given. Suppose that applying Algorithm 12 for all maps $\Pi, \Theta : 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$ we have that $\mathcal{P}_i \neq \emptyset$, $i \geq 2$ holds for some $n \geq \bar{n}$ and all $\bar{n} \in \mathbb{N}_0$ with $\mathcal{I}_p(n, 1) \neq \emptyset$ for some $p \in \mathcal{P}_i$. Then there exists no function $\beta \in \mathcal{KL}$ such that (14) holds for all $n \in \mathbb{N}_0$.*

Proof. Fix maps $\Pi, \Theta : 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$. Then Lemma 17 states that for each system $p \in \mathcal{P}_i$ there exists a system $q \in \mathcal{P}_j$, $j < i$ such that $q \in \mathcal{I}_p(n)$. If for any $p \in \mathcal{P}$ and any $n \in \mathbb{N}_0$ we have that $\mathbb{U}_p^{\text{ad}}(n, x_p(n), I_p(n)) = \emptyset$ or $\mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), I_p(n)) = \emptyset$ in case if cost functional (7) or (8) are used, respectively, we are done since no admissible solution exists. Otherwise, we obtain $u_p^{*1}(\cdot) \neq u_p^{*2}(\cdot)$ with

$$u_p^{*1}(\cdot) = \underset{u_p \in \mathbb{U}_p^{\text{ad}}(n, x_p(n), I_p(n))}{\operatorname{argmin}} J_p^{\infty}(x_p(n), u_p), \quad u_p^{*2}(\cdot) = \underset{u_p \in \mathbb{U}_p^{\text{ad}}(n, x_p(n), \emptyset)}{\operatorname{argmin}} J_p^{\infty}(x_p(n), u_p)$$

in case of cost functional (7) and with

$$u_p^{*1}(\cdot) = \underset{u_p \in \mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), I_p(n))}{\operatorname{argmin}} J_p^{N_p}(x_p(n), u_p), \quad u_p^{*2}(\cdot) = \underset{u_p \in \mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), \emptyset)}{\operatorname{argmin}} J_p^{N_p}(x_p(n), u_p)$$

in case of cost functional (8).

Hence, due to the fact that $x_p^{u_p^{*2}}(k, x(n))$ for some k violates a constraint imposed by system q which is not violated by $x_p^{u_p^{*1}}(k, x(n))$, we obtain that the open loop trajectories $x_p^{u_p^{*1}}(\cdot, x(n))$ and $x_p^{u_p^{*2}}(\cdot, x(n))$ differ. Using $\mathcal{I}_p(n, 1) \neq \emptyset$, we can conclude that there exists a $\delta_1 > 0$ such that $d_X(x_p^{u_p^{*1}}(1, x_p(n)), x_p^{u_p^{*2}}(1, x_p(n))) > \delta_1$ holds. Since we always implement the first element of each optimal admissible control, we have that $d_X(f_p(x_p(n), u_p^{*1}(0)), f_p(x_p(n), u_p^{*2}(0))) > \delta_1$ holds. Now we have to consider two cases: If $x_p(n+1) = x_p^{\text{ref}}$, then we can use the fact that the deviation $d_X(x_p^{u_p^{*1}}(1, x_p(\tilde{n})), x_p^{u_p^{*2}}(1, x_p(\tilde{n}))) > \delta_1$ will occur again for some $\tilde{n} > n$ due to the assumptions of the theorem. If $x_p(n+1) \neq x_p^{\text{ref}}$, we immediately obtain the existence of a $\delta_2 > 0$ such that $\|x_p(n+1)\|_{x_p^{\text{ref}}} > \delta_2$ holds. In either case, we obtain that there exists a time index $\tilde{n} > n$ such that $\|x_p(\tilde{n})\|_{x_p^{\text{ref}}} > \delta = \min(\delta_1/2, \delta_2)$ holds.

Now suppose there exists a function $\beta \in \mathcal{KL}$ such that (14) holds for all $n \in \mathbb{N}_0$. Due to the \mathcal{L} -property β in its second argument, we have that for each $\varepsilon > 0$ there

exists a $\hat{n} \in \mathbb{N}_0$ such that $\|x(n)\|_{x^{\text{ref}}} < \varepsilon$ for all $n \geq \hat{n}$. Now we choose $\varepsilon < \delta$ and $\hat{n} \in \mathbb{N}_0$ accordingly. Since Lemma 17 holds for all $\bar{n} \in \mathbb{N}_0$, we can conclude that for $\tilde{n} > n \geq \bar{n} = \hat{n}$ the inequality $\|x(\tilde{n})\|_{x^{\text{ref}}} \geq \|x_p(\tilde{n})\|_{x_p^{\text{ref}}} > \delta > \varepsilon$ holds. This contradicts the existence of a function $\beta \in \mathcal{KL}$ such that (14) holds for all $n \in \mathbb{N}_0$. Last, since the maps Π and Θ were chosen arbitrarily, the argumentation holds for all choices of Π and Θ which completes the proof. \square

Remark 19. Condition $\mathcal{I}_p(n, 1) \neq \emptyset$ in Theorem 18 is required since from $q \in \mathcal{I}_p(n)$ we can only conclude that $x_p^{u_p^{*1}}(k_n, x_p(n))$ and $x_p^{u_p^{*2}}(k_n, x_p(n))$ differ for some $k_n \geq 0$. According to the NMPC algorithm, only the first control element is implemented and we may face the situation that again $x_p^{u_p^{*1}}(k_{n+1}, x_p(n+1))$ and $x_p^{u_p^{*2}}(k_{n+1}, x_p(n+1))$ differ for some $k_{n+1} \geq k_n$. Now if $k_n > 0$ holds for all $n \in \mathbb{N}_0$, then system p may be asymptotically stable.

Turning from necessary to sufficient conditions we like to stress that the converse of Theorem 18 does not hold, not even in the special case that the conditions of Lemma 16 hold for all $n \geq \bar{n}$ with $\bar{n} \in \mathbb{N}_0$. This conclusion is due to the fact that even if $\mathcal{P}_2 = \emptyset$ we can only guarantee that a control which minimizes (8) for all systems $p \in \mathcal{P}$ can be computed without having to consider any other system $q \in \mathcal{P} \setminus \{p\}$, but not whether all systems are actually stable.

Theorem 20. Suppose that for given maps $\Pi, \Theta : 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$ we have that for a given initial value $x_0 \in \mathbb{X}$ the set of admissible controls $\mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), \mathcal{I}_p(n))$ is not empty for all $p \in \mathcal{P}$ and all $n \in \mathbb{N}_0$. Suppose furthermore that there exist $\alpha_1^p, \alpha_2^p, \alpha_3^p \in \mathcal{K}_\infty$, $\gamma \in \mathcal{K}_\infty$ and $\alpha > 0$ such that inequalities (16) and (17) hold for all $n \in \mathbb{N}_0$. Then there exists a function $\beta \in \mathcal{KL}$ which only depends on α , γ and all $\alpha_1^p, \alpha_2^p, \alpha_3^p$, $p \in \mathcal{P}$, such that (14) holds for all $n \in \mathbb{N}_0$.

Moreover, there exists an $\bar{n} \in \mathbb{N}_0$ such that for each $n \geq \bar{n}$ we either have that $\mathcal{P}_i \neq \emptyset$, $i \geq 2$ holds with $\mathcal{I}_p(n, 1) = \emptyset$ for all $p \in \mathcal{P}_i$ or $\mathcal{P}_2 = \emptyset$.

Proof. Using $x_0 \in X$, $\mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), \mathcal{I}_p(n)) \neq \emptyset$ for all $p \in \mathcal{P}$ and all $n \in \mathbb{N}_0$ and Theorem 13 we obtain that the closed loop solution $x(n) = (x_1(n)^\top, \dots, x_p(n)^\top)^\top$ exists for all $n \in \mathbb{N}_0$ and satisfies $x(n) \in \mathbb{X}$ for all $n \in \mathbb{N}_0$. Now, since inequalities (16) and (17) hold for all $n \in \mathbb{N}_0$, the existence of $\beta \in \mathcal{KL}$ follows directly from Proposition 10. To show the existence of $\bar{n} \in \mathbb{N}_0$ such that for $n \geq \bar{n}$ we either have that $\mathcal{P}_2 = \emptyset$ or $\mathcal{P}_i \neq \emptyset$, $i \geq 2$ holds with $\mathcal{I}_p(n, 1) = \emptyset$ for all $p \in \mathcal{P}_i$, suppose that $\mathcal{P}_i \neq \emptyset$, $i \geq 2$ holds for some $n \geq \bar{n}$ and all $\bar{n} \in \mathbb{N}_0$ with $\mathcal{I}_p(n, 1) \neq \emptyset$ for some $p \in \mathcal{P}_i$. Then, using Theorem 18 and the existence of $\beta \in \mathcal{KL}$ we obtain a contradiction showing the assertion. \square

Remark 21. While the stability result of Theorem 20 is given for the NMPC case without stabilizing terminal constraints or terminal costs, the only critical component in the proof of this theorem is the condition that $\mathbb{U}_p^{N_p, \text{ad}}(n, x_p(n), I_p(n)) \neq \emptyset$ which guarantees that the closed loop solution $x(\cdot) = (x_1(\cdot)^\top, \dots, x_p(\cdot)^\top)^\top$ exists and satisfies the state constraints. Hence, if instead of the existence conditions of $\alpha_1^p, \alpha_2^p, \alpha_3^p \in \mathcal{K}_\infty$ and $\alpha > 0$ such that inequalities (16) and (17) hold we impose other stability conditions – e.g., the terminal constraint condition given in Keerthi and Gilbert [15] or the terminal costs from Chen and Allgöwer [3] – then the same proof can be used to guarantee asymptotic stability of the closed loop.

5. Conclusion

We presented a generalized stability result for NMPC controllers without stabilizing terminal constraints or terminal costs. Moreover, we described an algorithm which allows us to generate a hierarchy of such controllers in a distributed non cooperative setting. Using only abstract priority and testing maps, we have shown necessary as well as sufficient conditions for stability of the closed loop.

Future research concerning the algorithm will certainly deal with the question how the priority and testing maps should be chosen to minimize the number of priority lists or to maximize the number of controllers that can be run in parallel. From the stability side an indeep analysis is required to apriori guarantee condition (17). The availability of such a condition would then allow us to apriori guarantee Algorithm 12 to asymptotically stabilize the system. One idea in this direction is outlined in Grüne and Worthmann [12, Section 7] and suggests the use of ISS small gain theorems to treat this problem.

References

- [1] Alamir, M., Bornard, G., 1995. Stability of a truncated infinite constrained receding horizon scheme: the general discrete nonlinear case. *Automatica* 31 (9), 1353–1356.
- [2] Bäckström, C., 1998. Computational aspects of reordering plans. *Journal of Artificial Intelligence Research* 9, 99–137.
- [3] Chen, H., Allgöwer, F., 1999. Nonlinear model predictive control schemes with guaranteed stability. In: *Nonlinear Model Based Process Control*. Kluwer Academic Publishers, Dodrecht, pp. 465–494.

- [4] Dold, J., Stursberg, O., 2009. Distributed Predictive Control of Communicating and Platooning Vehicles. In: Proceedings of the 48th IEEE Conference on Decision and Control held jointly with the 28th Chinese Control Conference CDC/CCC 2009. pp. 561–566.
- [5] Giselsson, P., 2010. Adaptive Nonlinear Model Predictive Control with Suboptimality and Stability Guarantees. In: Proceedings of the 49th Conference on Decision and Control. Atlanta, GA, pp. 3644–3649.
- [6] Giselsson, P., Rantzer, A., 2010. Distributed Model Predictive Control with Suboptimality and Stability Guarantees. In: Proceedings of the 49th Conference on Decision and Control. Atlanta, GA, USA.
- [7] Graichen, K., Kugi, A., nov. 2010. Stability and incremental improvement of suboptimal mpc without terminal constraints. Automatic Control, IEEE Transactions on 55 (11), 2576–2580.
- [8] Grimm, G., Messina, M., Tuna, S., Teel, A., 2005. Model predictive control: for want of a local control Lyapunov function, all is not lost. IEEE Trans. Automat. Control 50 (5), 546–558.
- [9] Grüne, L., Pannek, J., 2010. Analysis of unconstrained NMPC schemes with incomplete optimization. In: Proceedings of the 8th IFAC Symposium on Nonlinear Control Systems – NOLCOS 2010. Bologna, Italy, pp. 238–243.
- [10] Grüne, L., Pannek, J., 2011. Nonlinear Model Predictive Control: Theory and Algorithms, 1st Edition. Communications and Control Engineering. Springer.
- [11] Grüne, L., Pannek, J., Seehafer, M., Worthmann, K., 2010. Analysis of unconstrained nonlinear MPC schemes with varying control horizon. SIAM J. Control Optim. 48 (8), 4938–4962.
- [12] Grüne, L., Worthmann, K., 2011. A distributed NMPC scheme without stabilizing terminal constraints. In: Johansson, R., Rantzer, A. (Eds.), Distributed Decision Making and Control. Springer, 259–285.
- [13] Haupt, R., 1989. A survey of priority rule-based scheduling. Vol. 11. Springer Berlin / Heidelberg.

- [14] Jadbabaie, A., Hauser, J., 2005. On the stability of receding horizon control with a general terminal cost. *IEEE Trans. Automat. Control* 50 (5), 674–678.
- [15] Keerthi, S., Gilbert, E., 1988. Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations. *J. Optim. Theory Appl.* 57 (2), 265–293.
- [16] Lincoln, B., Rantzer, A., 2006. Relaxing dynamic programming. *IEEE Trans. Automat. Control* 51 (8), 1249–1260.
- [17] Rawlings, J. B., Mayne, D. Q., 2009. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, Madison.
- [18] Richards, A., How, J., 2004. A decentralized algorithm for robust constrained model predictive control. In: *Proceedings of the American Control Conference 2004*. Boston, Massachusetts, pp. 4261–4266.
- [19] Richards, A., How, J., 2007. Robust distributed model predictive control. *International Journal of Control* 80 (9), 1517–1531.
- [20] Scattolini, R., 2009. Architectures for distributed and hierarchical Model Predictive Control - A review. *Journal of Process Control* 19 (5), 723–731.